# 文件基本权限

rw-r--r--

rw- u所有者

r-- g所属组

r-- o其他人

r 读

w 写

x 执行

## chmod

修改文件权限：

chmod [选项] 模式 文件名

选项：

-R 递归

模式：

[ugoa] [+-=] [rwx]

[mode=421]

举例：

chmod u+x \*\*\*\*

chmod g+w,0+w \*\*\*\*

chmod a=rwx \*\*\*\*

## chown

## umask

查看默认权限

文件默认不能建立为执行文件，必须手动赋予执行权限，所以文件默认权限最大为666，默认权限需要换算成字母再相减，建立文件之后的默认权限，为666减去umask值。

# ACL权限

## 开启ACL权限

查看分区ACL权限是否开启：dumpe2fs -h /dev/sda1

dumpe2fs命令是查询指定分区详细文件系统信息的命令

选项：

-h 仅显示超级块中信息，而不显示磁盘块组的详细信息

临时开启分区ACL权限：mount -o remount,acl /

重新挂载根分区，并挂载加入ACL权限

永久开启分区ACL权限：

vi /etc/fstab

UUID=\*\*\*\* / ext4 defaults,acl 1 1（加入ACL）

mount -o remount /

重新挂载文件系统或重启动系统，使修改生效

## 查看ACL权限

指令：getfacl 文件名（查看ACL权限）

## 设定ACL权限

指令：setfacl 选项 文件名

选项：

-m 设定ACL权限

-x 删除指定的ACL权限

-b 删除所有的ACL权限

-d 设定默认ACL权限

-k 删除默认ACL权限

-R 递归设定ACL权限

给用户组设定ACL权限：

groupadd tgrp2

setfacl -m g:tgrp2:rwx /\*\*

为组tgrp2分配ACL权限，使用“g:组名:权限”格式。

## 最大有效权限mask

mask是用来指定最大有效权限的。如果我给用户赋予了ACL权限，是需要和mask的权限“相与”才能得到用户的真正权限。

## 删除ACL权限

指令：setfacl -x u:用户名 文件名（删除指定用户的ACL权限）

setfacl -x g:组名 文件名（删除指定用户组的ACL权限）

## 递归ACL权限

递归是父目录在设定ACL权限时，所有的子文件和子目录也会拥有相同的ACL权限。

指令：setfacl -m u:用户名:权限 -R 目录名

## 默认ACL权限

默认ACL权限的作用是如果给父目录设定了默认ACL权限，那么父目录中所有新建的子文件都会继承父目录的ACLL权限。

指令：setfacl -m d:u:用户名:权限 文件名

# 系统命令sudo权限

root把本来只能超级用户执行的命令赋予普通用户执行，sudo的操作对象是系统命令。

使用：visudo（实际修改的是/etc/sudoers文件）

root ALL=(ALL) ALL

#用户名 被管理主机的地址=（可使用的身份） 授权命令（绝对路径）

#%wheel ALL=（ALL） ALL

#%组名 被管理主机的地址=（可使用的身份） 授权命令（绝对路径）

举例：普通用户执行sudo赋予的命令

su - usr1

sudo -l（查看可用的sudo命令）

sudo /sbin/shutdown -r now（普通用户执行sudo赋予的命令）

举例：授予普通用户可以添加其他用户

viduso

-use1 ALL=/usr/sbin/useradd

-usr1 ALL=/usr/bin/passwd

授予用户设定密码的权限

# 文件特殊权限

## SetUID

SetUID作用：

1. 只有可以执行的二进制程序才能设定SUID权限
2. 命令执行者要对该程序拥有x（执行）权限
3. 命令执行者在执行该程序时获得该程序文件属主的身份（在执行程序的过程中灵魂附体为文件的属主）
4. SetUID权限只在该程序执行过程中有效，也就是说身份改变只在程序执行过程中有效

注：

passwd命令拥有SetUID权限，所以普通用户可以修改自己的密码。

cat命令没有SetUID权限，所以普通用户不能查看/ect/shadow文件内容。

设定SetUID的方法：

chmod 4755 文件名（4代表SUID）

chmod u+s 文件名

危险的SetUID：

1. 关键，目录应严格控制写权限，比如“/”、“/usr”等
2. 用户的密码设置要严格遵守密码三原则
3. 对系统中默认应该具有SetUID权限的文件作一列表，定时检查有没有这之外的文件被设置了SetUID权限

## SetGID

**SetGID针对文件的作用：**

1. 只有可执行的二进制程序才能设置SGID
2. 命令执行者要对该程序拥有x（执行）权限
3. 命令执行在执行程序的时候，组身份升级为该程序文件的属组
4. SetGID权限同样只在程序执行过程中有效，也就是说组身份改变只在程序执行过程中有效

**SetGID针对目录的作用：**

1. 铺平普通用户必须对此目录拥有r和x权限，才能进入此目录
2. 普通用户在此目录中的有效组会改变成此目录的属组
3. 若普通用户对此目录拥有w权限时，新建的文件的默认属组是这个目录的属组

举例：

1. /usr/bin/locate是可执行文件，可以赋予SGID
2. 执行用户lamp对/usr/bin/locate命令拥有执行权限
3. 执行/usr/bin/locate命令时。组身份会升级为slocate组，而slocate组对/var/lib/mlocate/mlocate.db数据库拥有r权限，所以普通用户可以使用locate命令查询mlocate.db数据库
4. 命令结束，lamp用户的组身份返回为lamp组

## Sticky BIT

SBIT黏着位作用：

1. 黏着位目前只对目录有效
2. 普通用户对该目录拥有w和x权限，即普通用户可以在此目录拥有写入权限
3. 如果没有黏着位，因为普通用户拥有w权限，所以可以删除此目录下所有文件，包括其他用户建立的文件。一旦赋予了黏着位，除了root可以删除所有文件，普通用户就算拥有w权限，也只能删除自己建立的文件，但是不能删除其他用户建立的文件。

# 文件系统属性chattr权限

格式：chattr [+-=] [选项] 文件或目录名

+：增加权限

-：删除权限

=：等于某权限

选项：i、a：i等价于insert，a等价于append

i：如果对文件设置i属性，那么不允许对文件进行删除、改名，也不能添加和修改数据；如果对目录设置i属性，那么只能修改目录下文件的数据，但不允许建立和删除文件。

a:如果对文件设置a属性，那么只能在文件中增加数据，但是不能删除也不能修改数据；如果对目录设置a属性，那么只允许在目录中建立和修改文件，但是不允许删除。